CLAIMS

What is claimed is:

1. A method, comprising:

selecting, during a compilation of code having one or more threads executable in a data processing system, a current thread having a most bottom order;

determining resources allocated to one or more child threads spawned from the current thread; and

allocating resources for the current thread in consideration of the resources allocated to the current thread's one or more child threads to avoid resource conflicts between the current thread and its one or more child threads.

2. The method of claim 1, wherein the resources include at least one of hardware registers and memory used by the respective thread.

3. The method of claim 1, wherein the resources allocated to the one or more child threads are recorded in a data structure accessible by the current thread.

4. The method of claim 1, further comprising updating resource information in a data structure regarding the resources allocated to the current thread, the data structure being accessible by a parent thread of the current thread.

5. The method of claim 1, further comprising repeating the selecting, determining, and allocating in a bottom-up order until each of the one or more threads has been processed.

6. The method of claim 5, further comprising allocate resources for a main thread that is a parent thread of the one or more threads after each of the one or more threads has been processed, the resources of the main thread are allocated in view of resources allocated to the one or more threads.

7. The method of claim 1, further comprising:

determining whether there are resources remaining in the data processing system prior to the allocating the resources for the current thread; and

deleting at least one child thread of the current thread; and

allocating the resources for the current thread using the resources associated with the at least one deleted child thread.

8. A machine-readable medium having executable code to cause a machine to perform a method, the method comprising:

selecting, during a compilation of code having one or more threads executable in a data processing system, a current thread having a most bottom order;

determining resources allocated to one or more child threads spawned from the current thread; and

allocating resources for the current thread in consideration of the resources allocated to the current thread's one or more child threads to avoid resource conflicts between the current thread and its one or more child threads.

9. The machine-readable medium of claim 8, wherein the resources include at least one of hardware registers and memory used by the respective thread.

10. The machine-readable medium of claim 8, wherein the resources allocated to the one or more child threads are recorded in a data structure accessible by the current thread.

11. The method of claim 1, further comprising updating resource information in a data structure regarding the resources allocated to the current thread, the data structure being accessible by a parent thread of the current thread.

12. The machine-readable medium of claim 8, wherein the method further comprises repeating the selecting, determining, and allocating in a bottom-up order until each of the one or more threads has been processed.

13. The machine-readable medium of claim 12, wherein the method further comprises allocating resources for a main thread that is a parent thread of the one or more threads after each of the one or more threads has been processed, the resources of the main thread are allocated in view of resources allocated to the one or more threads.

14. The machine-readable medium of claim 8, wherein the method further comprises:

determining whether there are resources remaining in the data processing system prior to the allocating the resources for the current thread; and

deleting at least one child thread of the current thread; and

allocating the resources for the current thread using the resources associated with the at least one deleted child thread.

15. A data processing system, comprising:

a processor capable of performing multi-threading operations;

a memory coupled to the processor; and

a process executed by the processor from the memory to cause the processor to

select, during a compilation of code having one or more threads executable in a data processing system, a current thread having a most bottom order,

determine resources allocated to one or more child threads spawned from the current thread, and

allocate resources for the current thread in consideration of the resources allocated to the current thread's one or more child threads to avoid resource conflicts between the current thread and its one or more child threads.

16. The data processing system of claim 15, wherein the process further causes the processor to update resource information in a data structure regarding the resources allocated to the current thread, the data structure being accessible by a parent thread of the current thread.

17. The data processing system of claim 16, wherein the process further causes the processor to repeat the selecting, determining, and allocating in a bottom-up order until each of the one or more threads has been processed.

18. The data processing system of claim 17, wherein the process further causes the processor to allocate resources for a main thread that is a parent thread of the one or more threads after each of the one or more threads has been processed, the resources of the main thread are allocated in view of resources allocated to the one or more threads.

19. The data processing system of claim 15, wherein the process further causes the processor to:

determine whether there are resources remaining in the data processing system prior to the allocating the resources for the current thread; and

delete at least one child thread of the current thread; and

allocate the resources for the current thread using the resources associated with the at least one deleted child thread.

20. The data processing system of claim 15, wherein the resources include at least one of hardware registers and memory used by the respective thread.